Chapter - 3
Research Methodology

Research methodology is the systematic and scientific approach used to conduct research, investigate problems, and gather data for a specific purpose. It involves techniques and procedures to identify, collect, analyze, and interpret data, addressing research questions or solving research problems

3.1 Significance of Research

Research methodology is the systematic approach to solving research problems, it involves selecting appropriate methods for data collection, analysis, and interpretation to ensure the validity and reliability of results. Key components include defining research questions, conducting literature reviews, choosing qualitative, quantitative, or mixed methods, and employing tools like surveys, experiments, or case studies. The proper methodology enables rigorous and reproducible findings, essential for advancing knowledge in their field. Understanding and applying the right methodology is crucial for producing high-quality, impactful research that withstands academic scrutiny.

The methodology ensures the research's validity, reliability, and reproducibility. Key aspects include selecting tools like surveys, experiments, or case studies, and applying statistical or thematic analysis techniques. A well-defined methodology is crucial for producing credible, high-quality research that contributes meaningfully to the academic field.

3.2 Research Gaps

Fog computing has attracted a great number of researchers so, it is a trending topic for research. The literature study motivates research in Fog Computing by introducing a bright future and its application of it. The researchers stated that Fog Computing will show how today's IoT and cloud computing work. The researchers also stated the challenges to be faced in the implementation of Fog Computing in real-life applications. Currently, researchers are working on the implementation of fog for commercial applications. The challenge for further studies and solutions from experts is that we need to keep ourselves updated for online publications and updates from the Open Fog consortium related to Fog Computing.

Even though fog computing has emerged as a potential standard paradigm that offers services to different IoT and mobile devices at the network edge, there are still many research issues that need to be resolved. Reaching the desired performance level, computing resource provisioning in terms of task offloading, and achieving the best response time with reduced latency are some examples of research challenges due to the heterogeneous nature of fog in terms of node capabilities while residing within the IoT domain.

Fog-Cloud Collaboration is a computing model that uses both fog and cloud computing. Fog computing processes data close to the source, reducing latency, while cloud computing handles large-scale data storage and processing. Together, they provide efficient, flexible data management, enhancing IoT performance, improving security, and supporting real-time applications. This collaboration optimizes resource usage, offering a scalable, sustainable solution for complex computing needs.

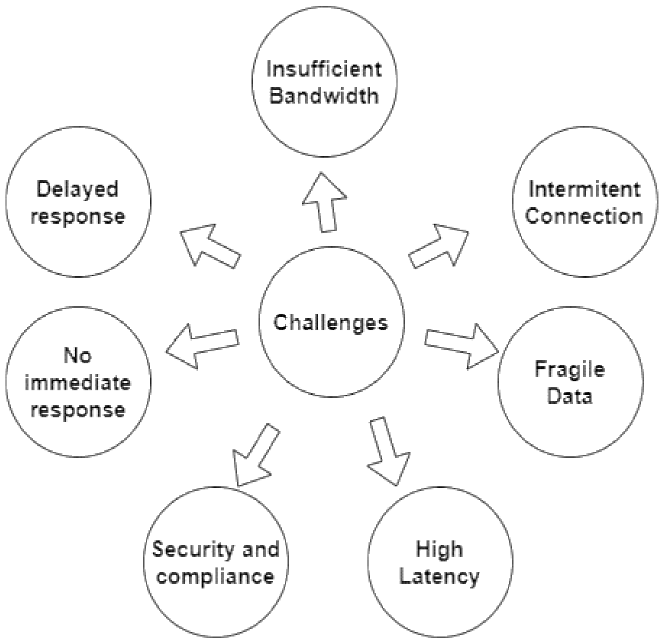


Figure 3.1: Data Processing Challenges at Cloud Data Center (Deafallah, 2022)

Figure 3.1 shows cloud data centers encounter several significant challenges in data processing due to the vast scale and complexity of their operations. Managing and processing big data from various sources requires robust distributed storage systems and parallel processing capabilities. Data security and privacy are crucial concerns, necessitating stringent access controls, encryption, and compliance with privacy regulations. Latency and network congestion can impact data processing performance, motivating the use of content delivery networks and edge computing strategies.

Scalability, resource allocation, energy efficiency, data backup, and disaster recovery planning are essential for maintaining optimal performance. Moreover, addressing data processing bottlenecks, handling heterogeneous data formats, complying with data privacy regulations from multiple jurisdictions, and enabling real-time data processing pose additional challenges. Cloud data centers continuously innovate and leverage advanced technologies to overcome these challenges, ensuring efficient and reliable data processing services for their users.

3.3 Problem Statement

Our research aims to understand the importance of cloud computing and fog computing. Fog computing solves problems like delay in response, insufficient bandwidth, no immediate response, security, and reduces the latency issue of cloud computing. The central problem focuses on:

"Smart Fog is a Collaborative Approach to Share Computational Power of Fog Devices for Fog Computing in Smart City IoT Network"

The research work studies the level of computational work, latency issue, and the efficiency of fog computational devices over various parameters like processing speed, scheduling, and task allocation in the fog layer, using fog computing and Machine Learning algorithms to reduce the problem and find trends, issue, challenges, suggestion, and future potential of computing problem in Fog Computing environment will share computational power to IoT devices with low computational power. Overall, the research work finds the use of fog computing networks to solve the future journey.

3.4 Objectives

The objectives of the research are clearly defined goals that guide the study, focusing on specific outcomes to be achieved. They include exploring new areas, describing phenomena, explaining relationships, predicting future events, and applying findings to solve real-world problems. Research objectives are specific goals that guide the focus and outcomes of a study. These objectives can vary widely but generally include exploration, description, explanation, prediction, application, evaluation, theory development, action, documentation, and innovation. These objectives ensure that the research remains focused, relevant, and systematic. They also help in evaluating the success and impact of the study. The purpose of this research work is to propose a "SMART FOG" protocol based on a technique to connect FOG computational devices which enables devices to share their resources within the Fog network and reduces the latency issue of cloud computing.

1. **To study IoT-based architectures and protocols for understanding the connectivity between IoT devices.**

2. **Analyse the current IoT infrastructure and evaluate various layers of communication protocols to design the SMART FOG Protocol-based technique.**

3. **Exploring the challenges to be faced in implementing the SMART FOG protocol-based technique on computation-enabled devices.**

4. **To explore the task scheduling and allocation techniques for Fog Computing nodes in SMART FOG.**

5. **Determine the fault tolerance mechanism in SMART FOG protocol-based technique by allocating tasks to multiple recipients.**

6. **Discover the efficiency of fog computational devices over various parameters like processing speed, scheduling, and task allocation in fog layer.**

Our research work focuses on the above-stated objective which aims to use the computational power of computation-enabled devices to collaboratively perform tasks and speed up the processing.

3.5 Hypothesis

The hypothesis is nothing but a tentative statement to predict the expected outcomes of a study. Defining hypothesis helps in designing new experiments and observations. The following hypotheses were tested for the system in the proposed research work.

This hypothesis is subdivided into $H_0$1 to $H_0$6 to explore the efficiency and various measures of the Smart Fog protocol-based system compared to cloud-based systems, aiming to comprehensively evaluate their impact and effectiveness. The sub hypothesis from H_a1 to H_a6 are as follows:

1. **$H_0$1:** There is no significant difference between SMART FOG protocol-based system and cloud-based system based on the performance measure execution time.

2 **$H_0$2:** There is no significant difference between SMART FOG protocol-based System and cloud-based system based on the performance measure latency.

3 **$H_0$3:** There is no significant difference between SMART FOG protocol-based system and cloud-based system based on the performance measure energy consumed.

4 **$H_0$4:** There is no significant difference between SMART FOG protocol-based system and cloud-based system based on the performance measure cost of execution.

5 **$H_0$5:** There is no significant difference between SMART FOG protocol-based system and cloud-based system based on the performance measure of total network usage.

6 **$H_0$6:** There is no significant difference between SMART FOG protocol-based system and cloud-based system based on the performance measure computational power consumed.

Dividing the main hypothesis into these sub-hypotheses enables researchers to methodically investigate different facets of the Smart Fog protocol-based system and compare its efficiency and performance against a cloud-based counterpart. This methodical approach facilitates a thorough assessment of Fog Computing technology and its potential benefits in comparison to traditional cloud solutions within IoT environments.

3.6 Scope of Study

As studying about advances in new computational paradigm and the use of cloud computing and IoT have great futuristic applications. This emerging IoT introduces many challenges which cannot be handled by today's cloud computing. In this research work we deal with the IoT Environment features like low latency, high distribution, large-scale sensor network, and mobility support and device heterogeneity. This proposed SMART FOG system allows us to create a collaborative environment for IoT network. In the proposed system, we are going to implement SMART FOG protocol-based technique which will allow Fog nodes to share computing and storage power to IoT devices that have low computational power within IoT network. The proposed system will be able to schedule the tasks assigned to fog node for easy processing and efficient resource management. The proposed work is focused on creating a resilient environment using SMART FOG which will create trust in fog computing. As fog computing is in its infancy, there are still many open challenges are present. The SMART FOG will create trust between fog clients and fog environment by providing fault-tolerant and secure technique for fog computing. This research will identify some of these challenges and try to find a solution in the proposed system.

3.7 Research Methodology

The research methodology deals with the hypothesis which is the outcome of the objectives with the results. The proposed study attempts to implement SMART FOG, a collaborative approach using Fog Computing. The research involves quantitative and qualitative approaches. The SMART FOG collaboratively used computational power and storage of devices connected within Fog layer. The fog layer acts as an intermediate between IoT networks and Cloud centers. In this research, we have identified different open challenges in fog computing and tried to resolve some of them using SMART FOG. The hypothesis is nothing but a tentative statement to predict the expected outcomes of a study. Defining hypotheses helps in designing new experiments and observations. The following hypothesis is being tested for the system in the proposed research work. The hypothesis has arrived at the expected outcome of the system. "SMART FOG protocol-based technique to create Fog Computing environment will share computational power to IoT devices with low computational power".

3.7.1 Sources of Information

Information is gathered from journals, articles, and publications about fog computing are the main sources of information. The OpenFog consortium provided white papers which are very useful as the main source of information. The OpenFog community provided papers on the taxonomy of Fog Computing, basic architectural design, and structure of fog computing and their multiple layers of implementation. For further information, various articles from industrial experts who are connected to Fog Computing will act as a lighthouse in the dark.

3.7.2 Data Collection

The universe for the present study is comprised of smart city applications, all peoples within a smart city, and cloud centers. The universe also includes the devices within the IoT network, fog devices, etc. The smart city sample application is being randomly selected to implement SMART FOG and the overall performance and efficiency are being evaluated.

As per the SMART FOG applications, the IoT network generates some amount of data using sensors that are being used in computing and to test the performance of SMART FOG. The data generated in this research work is application-oriented. If we consider security surveillance applications for smart cities then the data is collected of images of events, video streams collected from cameras, and being used for further applications. The main input to the proposed system is requests from IoT devices for shared computational power.

Cloud -Fog Simulators for Data Collection

The primary goal of the research work is to examine and find the technologies associated with the SMART FOG project. To find new emerging technologies that can impact the cloud system in SMART FOG computing and also improve the reliability of Predictive models based on Artificial Intelligence and Machine Learning Algorithms is being developed to resolve computing problems.

iFogSim Simulators

Many available simulators can simulate the scenarios of cloud-fog computing environments such as EdgeCloudSim, MobIoTSim, SimpleIoTSimulator, IBM BlueMix, Google IoT Sim, and EmuFog. Most of the available simulators are similar in their functionalities, programming language, or architecture. Therefore, we limited our study to only eight main simulators. The simulators are analyzed both from theoretical and practical perspectives. In theoretical comparisons, all eight simulators (iFogSim, iFogSim2, FogNetSim++, EdgeCloudSim, FogComputingSim, PureEdgeSim, YAFS, and LEAF) are compared based on their technical and non-technical characteristics, whereas for practical comparison use iFogSim, are in terms of their execution time, memory usage, and CPU consumption for simulating different applications under varying complexities.

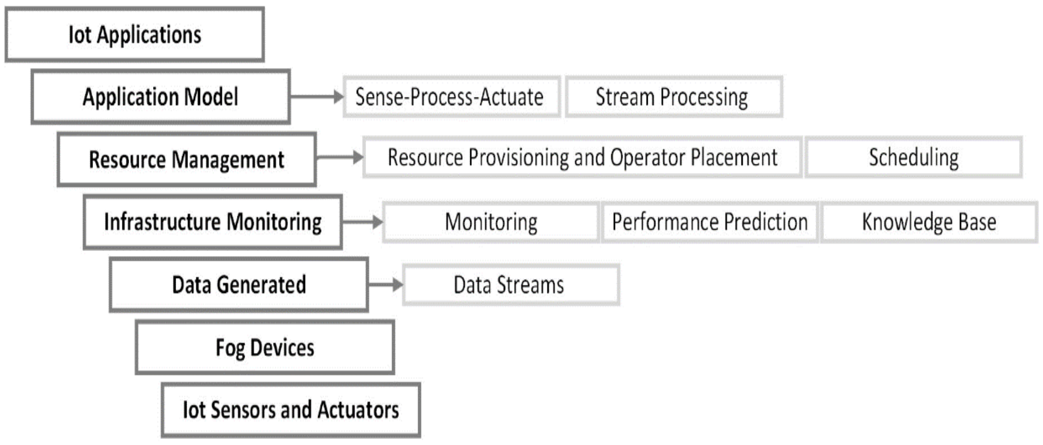


Figure 3.2: iFogSim Architecture (Muhammad, 2023)

The iFogSimToolkit provides a platform for modeling and simulation of resource management techniques in edge, Fog computing, and cloud environments. A newer version of iFogSim, adds distributed clustering, mobility, and microservices management as new features. Furthermore, it includes new example scenarios to validate and demonstrate their extension for the iFogSim. the architecture used by iFogSim is shown in Figure 3.2.

Due to the IoT revolution, almost everything is becoming a source for data generation. As a result, a tremendous amount of data is generated every second. Huge amount of data processed on workstations. typical data sources include mobiles, various types of sensors and actuators including thermostats, engines of airplanes, factories, mobiles, computers, automobiles such as driverless cars, metros, human health data, smart devices such as Google Home, Alexa Echo Dots, smart homes, smart shoes, watches, and, in general, all wearables, etc., and the number of items on the list increases all of the time. These data need to be pre-processed before something useful can be derived from them because only some of the generated data are relevant or useful. This section will look at the various sources from which data is generated.

3.7.3 Through Participation in Conference and Paper Published

To collect insight into the subject and dive deeper into the details of fog computing, Cloud Computing, Artificial Intelligence, and Machine Learning algorithms following conferences were attended.

a) International Virtual Conference "Emerging Era of Applications of Computer: The Survey on Fog Computing and its Applications" on 15th - 16th of January 2022 Organized by Pacific University Udaipur.

b) "Use of Clustering Machine Learning Algorithms in Fog Computing for Task Scheduling and Resource Allocation" has been published in European Chemical Bulletin (ISSN: 2063-5346), Volume 11, Issue 8, 2022 Date of Publication: - August 2022.

c) National Seminar on "Implementation of Academic Bank of Credit (ABC) in Higher Education Institutes" on 21st March 2023 Organized by Avinashilingam Institute for Home Science and Higher Education for Women University Udaipur.

d) IP Awareness Training Program under "National Intellectual Property Awareness Mission" Organized by Intellectual Property Office, India on 18, January 2023.

e) "A Comparative Study of Various Classification Machine Learning Algorithms in Fog Computing: Task Scheduling" has been published in Industrial Engineering Journal (ISSN 0970-2555), Volume: 52, Issue 5, No. UGC Care Approved, Group I, Peer Reviewed Journal 4, May: 2023.

I am grateful to all Conference Organizers and my fellow presenters and researchers who not only provided me with the platform to showcase my talent but also helped me with rich technical experience by actively participating in a conference to collect data. These gatherings have provided me the stage for scholarly exchange which helped me a lot in coming out with Machine learning solutions for traffic congestion problems.

3.7.4 Performance Evaluation

The performance of the developed machine learning predictive model is analyzed using various performance measures such as prediction accuracy, incorrectly classified instances, kappa score, and various confusion matrix parameters such as true positive rate, false positive rate, precision, recall, and F1-score. Compare the performance of the model with existing traffic prediction models and assess its effectiveness in predicting traffic congestion and optimizing transportation systems.

3.7.5 Machine Learning Predictive Model Development

Designed and developed a machine learning predictive model for smart fog systems using the gathered data and insights from the literature review. Utilize appropriate machine learning algorithms such as regression, Random Forest, Random Tree, Bayes net, naïve Bays, SMO, IBK, Logistic Regression, K-Star, and Multiclass classifier in addressing cloud issues.

3.8 Tools and Technique

A method known as SMART FOG uses nearby fog nodes to complete tasks to utilize cloud centers less frequently and with less delay. Therefore, to put this system into place, firstly build an IoT network and cloud application that can handle requests from the IoT network and store the data on servers. After that, an interface protocol is created to essentially connect the cloud and Internet of Things network and process requests that he can process rather than sending them to the cloud. Finally, the IoT network receives the results.

In the proposed system, the requesting IoT device can use the publish method to submit a request to the closest fog devices, and the nearest fog device that is available will accept the request and subscribe to share computing power. A few further security precautions are required and are being implemented in SMART FOG to safeguard the connection to prevent attacks like Man-In-the-Middle or identity theft.

Fog computing requires various tools, techniques, and algorithms to optimize data processing and management close to the network edge. Key components include Network Management Software-defined networking tools that optimize traffic routing and resource allocation. Virtualization technology tools like Docker deploy applications in isolated environments efficiently Data Analytics Real-time analytics tools, such as Apache Kafka and Apache Storm, process data streams at the fog layer, Weka 3.8.6. Machine Learning Algorithms like decision trees, K-Nearest Neighbor, Logistic regression, K-Star, IBK, J48, Bagging, MLP clustering, and neural networks analyze and predict data trends locally, and Resource Management Algorithms include load balancing and task scheduling algorithms to optimize resource utilization and performance. These tools and techniques collectively enhance the efficiency, scalability, and security of fog computing systems.

To evaluate the performance and effectiveness of the smart fog systems, various metrics and statistical methods were employed. Percentage analysis, measures of central tendency, measures of dispersion, cumulative frequency, correlation coefficient, and regression analysis were used to analyze the collected mining data. Hypothesis testing was performed using the Chi-Square test. The study involved the simulation of data and model building, utilizing multiple regression analysis. A conceptual model based on regression was developed to examine the significance of different technologies. Additionally, the study aimed to assess the usefulness of loT, Artificial Intelligence, and Machine Learning-based models in addressing commuting problems. The Weka tool and Python were used for simulation and predictive analysis. Overall, the study employed a research design that combined qualitative and quantitative research approaches. The qualitative nature of the study facilitated the exploration of various concepts and ideas, leading to findings and recommendations for improving fog computation.

3.8.1 Weka Tools Techniques

The Weka Experimenter is a tool within the Weka software package that allows users to design, run, and analyze machine learning experiments systematically. It is particularly useful for comparing multiple machine learning algorithms and configurations on various datasets, helping researchers and practitioners make informed decisions about which algorithms work best for their specific tasks. Here’s a more detailed explanation of the Weka Experimenter’s key features and functionalities.

Experiment Design: The Experimenter allows users to design experiments by specifying different machine learning algorithms, datasets, and evaluation metrics. Users can choose from a wide range of classification, regression, and clustering. Algorithms available in Weka. They can also select multiple datasets to test the algorithms' performance across different data domains.

Parameter Sweeping: Users can explore the effect of different parameter settings on the performance of machine learning algorithms. The Experimenter enables parameter sweeping, where users can specify a range of values for certain parameters of the algorithms. The Experimenter then systematically runs experiments with different parameter combinations to find the optimal settings.

Cross-Validation and Evaluation Metrics: The Experimenter supports various techniques for evaluating machine learning models, including cross-validation (k-fold cross-validation, leave-one-out cross-validation, etc.). Users can select different evaluation metrics such as accuracy, precision, recall, F1-score, and others to assess the performance of the algorithms.

Batch Execution: The Experimenter can run experiments in batch mode, allowing users to schedule multiple experiments to run sequentially or concurrently. This feature is particularly useful for running large-scale experiments overnight or on computing clusters.

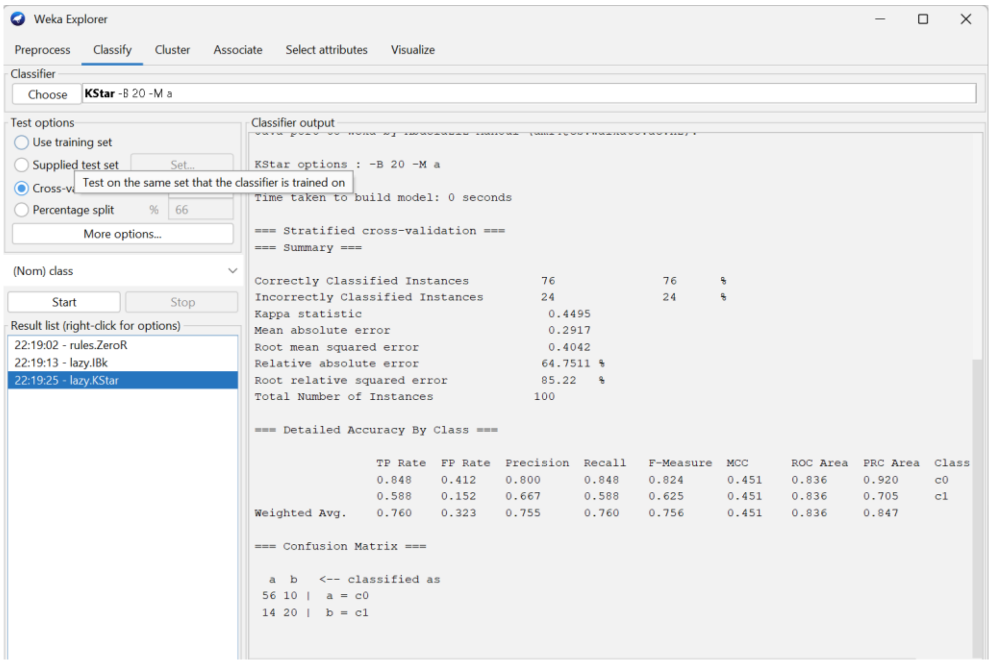
Result Analysis and Comparison: After the experiments are completed, the Experimenter provides detailed summary reports and visualizations of the results. Users can compare the performance of different algorithms on various datasets using statistical tests and visualizations like charts and graphs. This comparative analysis helps users identify the best-performing algorithms and configurations for their specific problem domains.

Reproducibility: The Experimenter ensures the reproducibility of experiments by allowing users to save the experiment configurations and results. Researchers can share these configurations and results with others, making it easier to validate and replicate experiments.

Integration with Other Weka Tools: The Experimenter seamlessly integrates with other Weka tools and interfaces, allowing users to utilize preprocessing techniques, attribute selection methods, and various machine learning algorithms available in Weka.

The Weka Experimenter provides a user-friendly environment for designing, and running. And analyzing machine learning experiments. Its capabilities make it a valuable tool for researchers and practitioners who want to systematically evaluate and compare different machine-learning algorithms and configurations on multiple datasets. Some key tools and techniques available in Weka Explorer include:

- Preprocessing Tools
- Classification Algorithms
- Clustering Algorithms
- Attribute Selection
- Evaluation Techniques
- Visualization Tools



KStar -B 20 -M a

(Nom) class

% 66

22:19:02 - rules.ZeroR
22:19:13 - lazy.IBk
22:19:25 - lazy.KStar

```
KStar options : -B 20 -M a

Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          76               76      %
Incorrectly Classified Instances        24               24      %
Kappa statistic                          0.4495
Mean absolute error                      0.2917
Root mean squared error                  0.4042
Relative absolute error                 64.7511 %
Root relative squared error             85.22   %
Total Number of Instances              100

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.848    0.412    0.800      0.848    0.824      0.451    0.836     0.920     c0
                 0.588    0.152    0.667      0.588    0.625      0.451    0.836     0.705     c1
Weighted Avg.    0.760    0.323    0.755      0.760    0.756      0.451    0.836     0.847

=== Confusion Matrix ===

  a  b   <-- classified as
 56 10 |  a = c0
 14 20 |  b = c1
```

Figure 3.3: Weka Tool K-Star

Figure 3.3 shows the preprocessing tool in Weka applied to This interface includes details about the number of instances, number of attributes, relation, selected attribute tab, etc. In the present scenario, the details of the Time attribute are shown in the selected attributes tab.

3.8.2 Experimental Setup

CCTV applications process data recorded by cameras deployed at STL[1]. The source task, located at the STL, sends 10 Mbps of video data to a processing task that requires 30,000 MIPS and is responsible for traffic monitoring, enforcing traffic laws, and automatic incident detection. The 200 kbit/s of resulting data are sent to the sink task located in the cloud for further analysis and storage. 16 of these applications are running in our scenario, one for each STL.

Although the computational and network load required by the CCTV applications is constant during the entire simulation, the reported power consumption varies over time. This is because we allocate the static power consumption of fog nodes proportionally to applications running on them and fog nodes are utilized inefficiently in this experiment. Especially at night when only a few taxis are on the road, the relative power demanded by the CCTV applications rises.

3.8.3 Hypothesis Testing Tool

To test the famed null hypothesis, three types of statistical methods were used. The applied tests were the Pearson Chi-Square test, ANNOVA Test, and T-Test.

Chi-Square Test:

The Chi-Square test is a statistical method used to determine if there is a significant association between categorical variables. It compares observed frequencies with expected frequencies, assessing whether any differences are statistically significant.

The Formula for Chi Square Is

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

where:

$c =$ degrees of freedom

$O =$ observed value(s)

$E =$ expected value(s)

[1] Seasonal and Trend decomposition using Loess

3.9 Applied Methodology

In this section, the proposed model, and the interaction among its components with the essential interfacing requirements are demonstrated. The proposed model consists of three layers concerning intelligent task offloading in fog cloud systems. It is composed of both fog and cloud servers. The underlying fog-cloud environment is comprised of distributed resources that are heterogeneous in terms of network hierarchy starting from the very basic physical layer of a network to the centralized cloud environment. Heterogeneous means these devices are dispersed at different geolocations and not stationary. The host servers, which perform as computing resources, intended for providing services to various application tasks, are enriched with a diverse set of resources. It is based on two types of applications, i.e., delay-sensitive applications and computation-intensive applications.

3.9.1 Fog-Cloud Smart Task Offloading Model

Mainly the architecture includes three layers with a smart task offloading management system which includes predictive and prescriptive constructs as shown in the figure below. The three layers included are the IoT or physical layer, Fog node layer, and the Cloud layer.

Offloading Management System is an intelligent framework designed to optimize task offloading decisions in distributed computing environments, particularly in fog computing and edge computing systems. The starting point in a task offloading procedure possess five main features:

1) Policy Repository regarding Offloading criteria
2) Recent status of fog snapshot
3) Receive, analyse, and offload the tasks
4) Prediction construct
5) Prescriptive construct

The entire process is activated by the Smart OMS. Its formation consists of a Policy Repository regarding Offloading criteria, monitoring & organizing offloading procedures, a recent snapshot of fog competence & readiness, a Prediction construct, and a Prescriptive construct.

3.9.2 Task Offloading

The volatile demand from IoT and mobile devices, which may not be predicted or anticipated immediately due to the unpredictability of the fog resources, the issue has to be handled. The main goal of this study is to provide a predictive and prescriptive approach for cost-effective task offloading and resource scheduling that will maximize the cost of these devices executing their applications. As a result, this research work cover addresses the mentioned issues and offers suggestions for how to fix them. Furthermore, some tasks share the same fog resources; as a result, there may be resource conflicts in some situations that could result in deadlock, some tasks experience delayed responses, and it's possible that new tasks won't be able to acquire resources at all, which is where latency comes into play. To ensure the equitable use of the underlying resources, it must be decided to improve fog performance by offloading some activities to adjacent nodes.

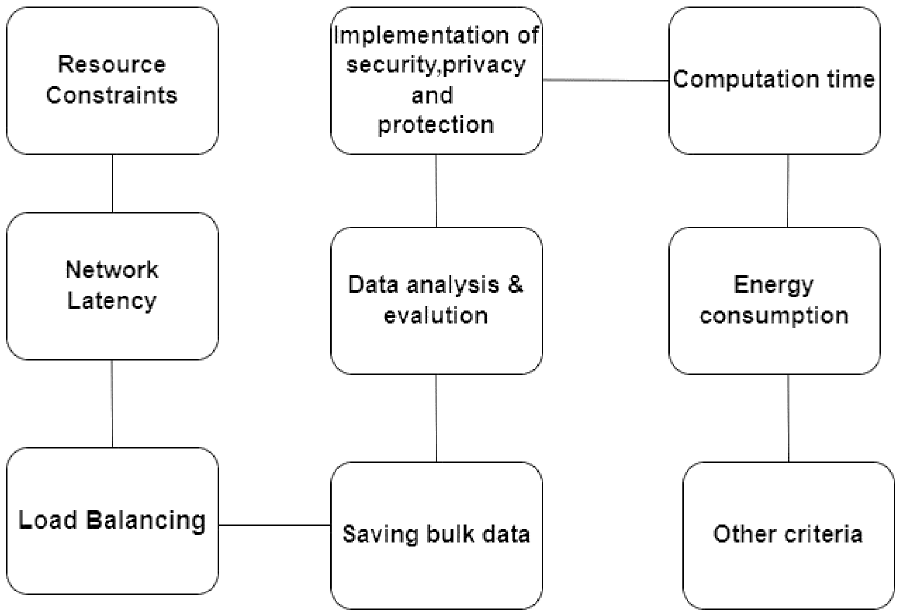


Figure 3.4: Task offloading Criteria (Satyakam, 2021)

Figure 3.4, shows Task offloading is a crucial process in fog computing and edge computing environments, where computational tasks are transferred from resource-constrained IoT devices to more capable fog nodes or cloud servers.

The decision-making process for task offloading considers various conditions, such as resource constraints, network latency, load balancing, security, privacy, data evaluation, storing bulk data, execution time, energy consumption, and other specific criteria. By balancing these factors, task offloading aims to optimize resource utilization, reduce latency, improve energy efficiency, and enhance overall system performance. Offloading computationally intensive tasks to more powerful nodes, considering network conditions, and ensuring data privacy and security play key roles in achieving efficient and effective task offloading in distributed computing systems.

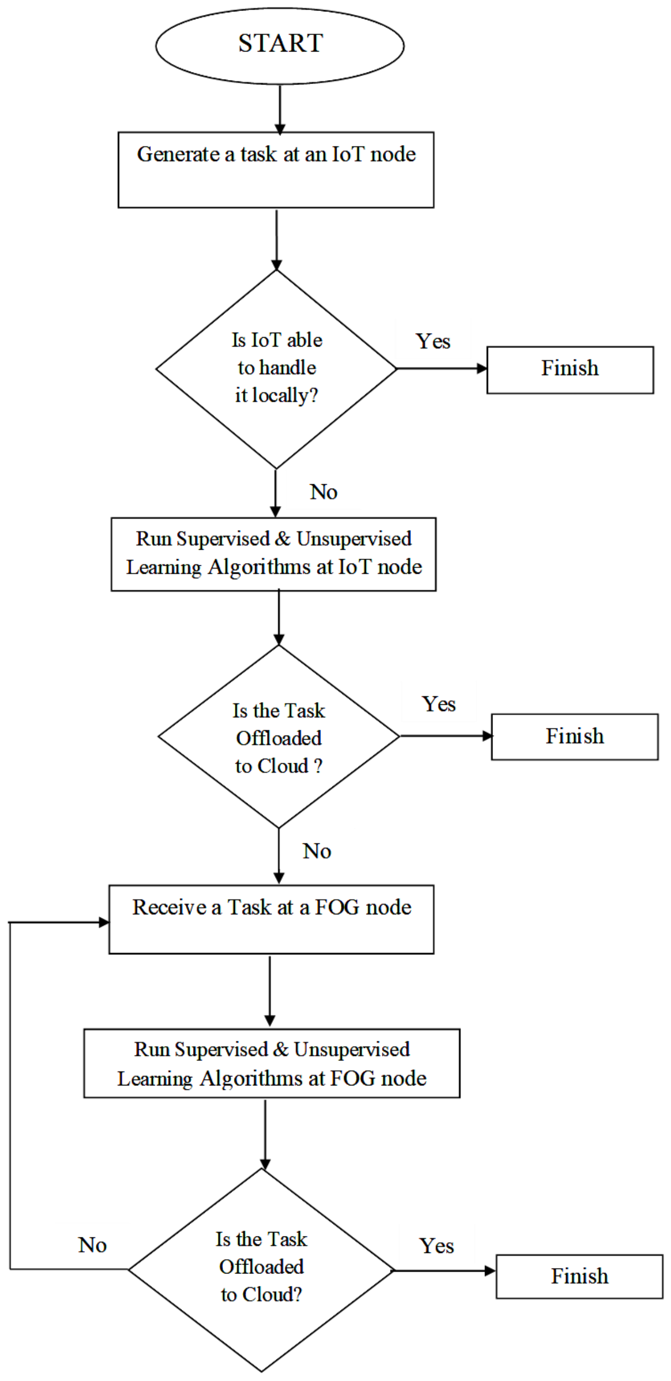

START
Generate a task at an IoT node
Is IoT able to handle it locally?
Yes
Finish
No
Run Supervised & Unsupervised Learning Algorithms at IoT node
Is the Task Offloaded to Cloud ?
Yes
Finish
No
Receive a Task at a FOG node
Run Supervised & Unsupervised Learning Algorithms at FOG node
No
Is the Task Offloaded to Cloud?
Yes
Finish

Figure 3.5: Flow Diagram: SMART FOG Task Offloading (Li, 2019)

Figure 3.5 shows the various steps used for the task offloading as shown above in the figure. In the proposed flowchart, a task is generated on an IoT node, and the node evaluates its capability to execute machine learning algorithms locally. If the IoT node lacks resources or the task complexity exceeds its capacity, the task is offloaded to the cloud. Alternatively, if the IoT node can handle the task, it executes the machine learning algorithms. The task is then sent to a fog node, which assesses its resources and executes machine-learning algorithms like classification and clustering. If the fog node is unable to handle the task, it may offload it back to the cloud. After the completion of tasks, results are delivered to the IoT node or end-user, based on application requirements and data privacy considerations. This dynamic process ensures optimal resource utilization, reduced latency, and enhanced performance in the IoT and fog computing environments.

3.9.3 Workflow Diagram

workflow diagram illustrating the progression of research and development in the field of SMART FOG. The workflow diagram illustrates the progression of research and development in the field of SMART FOG. It starts with understanding cloud, IoTs, and fog computing concepts and then delves into analyzing IoT-based architectures and protocols. Next, the focus shifts to evaluating various layers of communication protocols and devising improved task scheduling and allocation techniques for fog computing nodes.

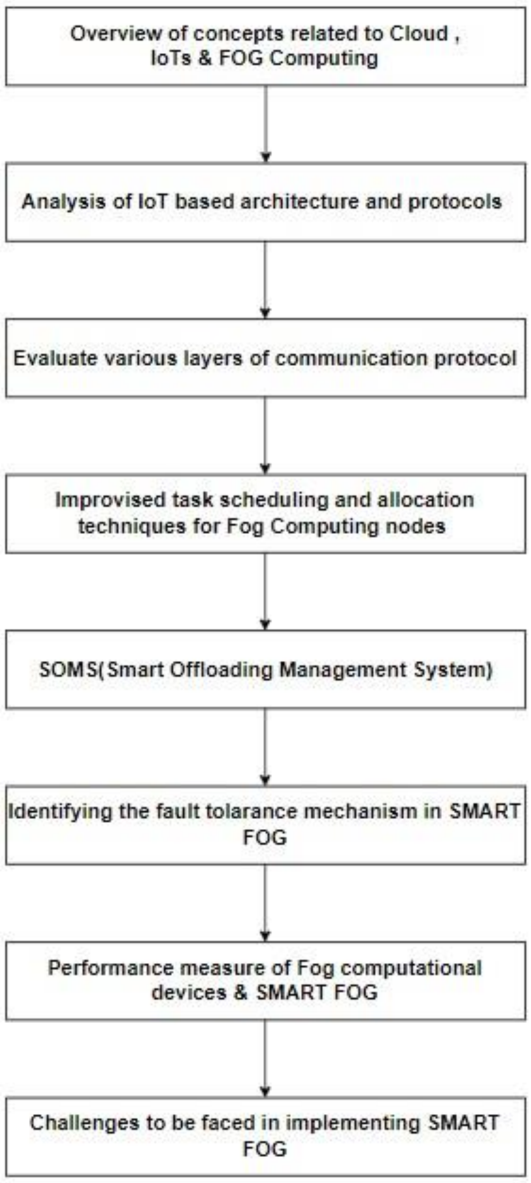

Overview of concepts related to Cloud , IoTs & FOG Computing
Analysis of IoT based architecture and protocols
Evaluate various layers of communication protocol
Improvised task scheduling and allocation techniques for Fog Computing nodes
SOMS(Smart Offloading Management System)
Identifying the fault tolarance mechanism in SMART FOG
Performance measure of Fog computational devices & SMART FOG
Challenges to be faced in implementing SMART FOG

Figure 3.6: Workflow Diagram

Figure 3.6 shows depict the implementation and functioning of the Smart Offloading Management System, which optimizes task offloading decisions. It also highlights the identification and incorporation of fault tolerance mechanisms in SMART FOG to ensure system reliability. Furthermore, the diagram emphasizes the importance of measuring the performance of fog computational devices and the SMART FOG system. Finally, the challenges faced during the implementation of SMART FOG are outlined, underscoring the need to overcome hurdles to achieve successful deployment and operation.

3.10 Performance Metrics for Supervised and Unsupervised Algorithms

Performance metrics for supervised algorithms include accuracy, precision, recall, F1-score, and area under the ROC curve, which assess the model's prediction quality. For unsupervised algorithms, metrics like silhouette score, Davies-Bouldin index, and clustering accuracy evaluate the coherence and separation of clusters. These metrics help determine how well the algorithms perform their respective tasks in various data analysis scenarios.

According to Figure 3.7, various studies to find the quality and performance of the various clustering algorithms various measures are being suggested but finding one is a challenging task in unsupervised learning. Some of the major performance evaluation clustering methods or clustering validity indexes can be classified as external, internal, and relative as shown in the figure below.

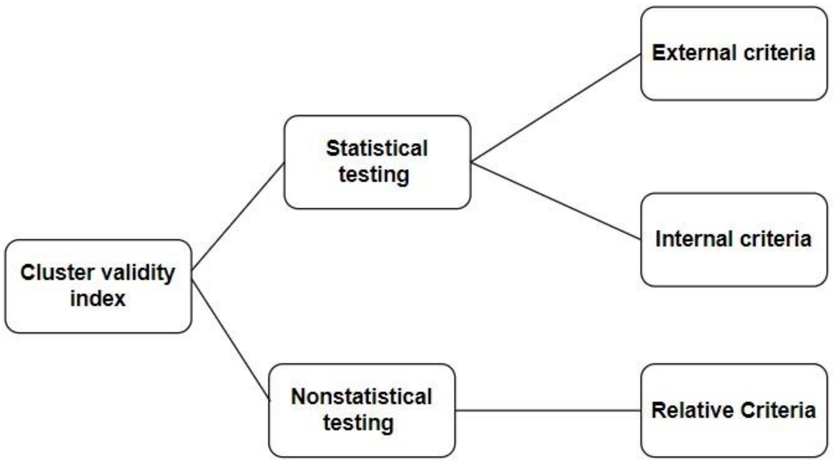


Figure 3.7: Cluster validity index (Wang, 2019)

3.10.1 Internal Validation

Figure 3.8 shows internal validation criteria are being used when we are not having additional information about the datasets. In such cases, the quality of the clustering algorithm can be measured by the two basic approaches partitioned and Hierarchical.

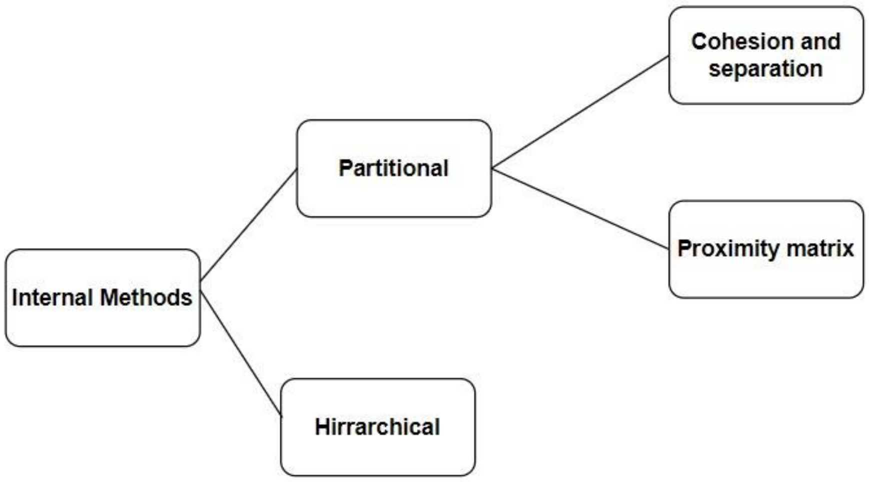


Figure 3.8: Internal Validation Method (Wang, 2019)

In situations where external information or ground truth labels are unavailable, internal validation criteria play a crucial role in assessing the quality of clustering algorithms. These methods evaluate clustering results based solely on the data's characteristics and clustering structure.

3.10.2 External Validation

External validation methods are considered with the supervised learning or classification problems. External validation methods can be also incorporated if additional information or class labels are available in the particular clustering problems that have the class labels for the training sets. For applying external validation various aspects are to be taken into consideration which are as follows

- Required to find clustering tendency for a particular dataset
- Find the correct number of clusters.
- Use internal methods for measuring the quality of clusters first.
- Now compare the internal method results with the external information.
- Make a comparison between the two sets of clusters to find the best one.

Figure 3.9 shows to find the clustering tendency for a given dataset, internal clustering validation methods are utilized to measure the quality of clusters without relying on external information. These methods, such as the Silhouette Score, Davies-Bouldin Index, and Dunn Index, help identify the correct number of clusters that yield the highest quality results. By comparing the internal validation outcomes with external information, obtained through external validation methods like Adjusted

Rand Index or Normalized Mutual Information when available, the clustering results can be evaluated against known class labels or ground truth data. The best clustering solution is determined by considering both internal and external validation results, aiming to achieve consistency and high-quality clusters.

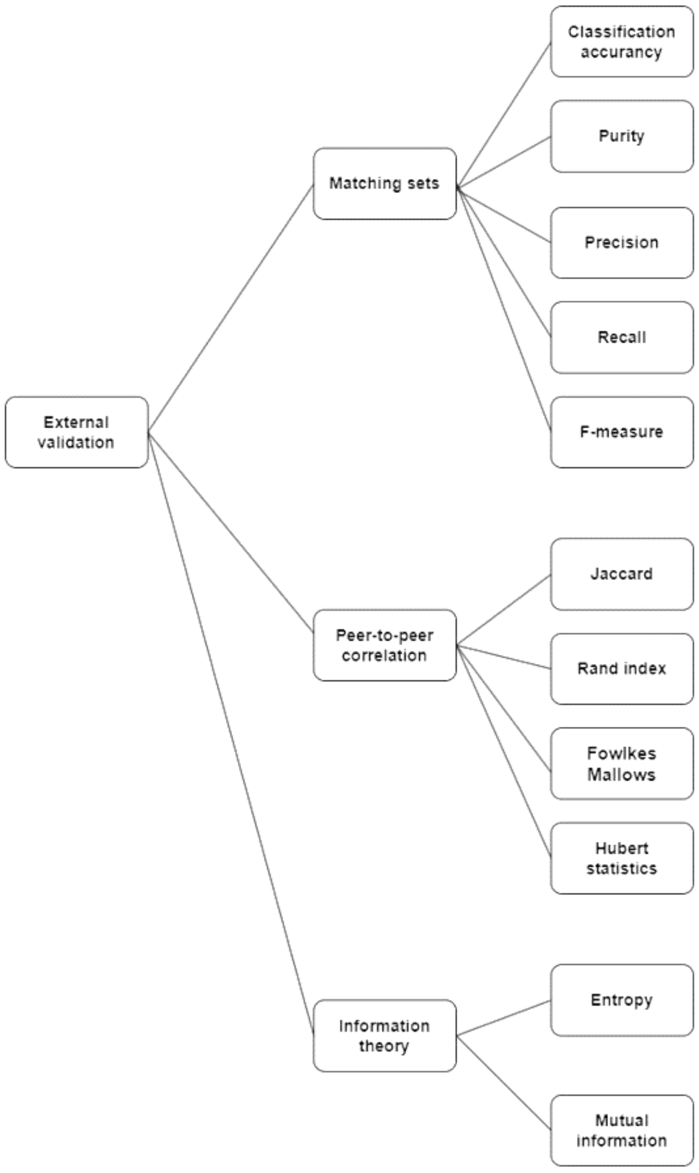


Figure 3.9: External Validation Method (Wang, 2019)

The external criteria are applied as in the clustering algorithm suppose C = {C1, C2…...Cm} represent the clustered partition and P = {P1, P2…. Ps} represent the true partition obtained from expert knowledge or class labels.

TP[2]: The no. of data points found in the same particular cluster, both C and P.

[2] True Positive

FP[3]: The no. of data points found in the same particular cluster in C but in a different cluster.

FN[4]: The no. of data points found in different clusters in C but in the same cluster in P.

TN[5]: The no. of data points found in different clusters, both in C and in P.

The no. of data points found in the same cluster in C:

m1 = TP + FP.

The no. of data points found in the same cluster in P:

m2 = TP + FN.

M = TP + FP + FN + TN.

These external validation methods help assess the accuracy, consistency, and robustness of clustering algorithms by comparing their results with known ground truth information or externally provided criteria. By utilizing these validation techniques, researchers and practitioners can make informed decisions about the suitability and performance of clustering algorithms for their specific applications and datasets.

Matching Sets

The first category in external criteria includes the measuring parameters like recall, precision, TP, TN, FP, FN, error, F- measure, etc. Precision can be calculated by number of the true positives

$$Pr = \frac{TP}{TP + FP} = \frac{TP}{P} = \frac{p_{ij}}{p_i}$$

Recall measures the percentage of data points properly included in the same particular cluster:

$$R = \frac{TP}{TP + FN} = \frac{p_{ij}}{p_j}$$

The F-measure is a combination of precision and recall

[3] False Positive
[4] False Negative
[5] True Negative

$$F_\alpha = \frac{1+\alpha}{\frac{1}{Pr} + \frac{\alpha}{R}}$$

The F-measure, also known as the F1 score, is a metric used to evaluate the accuracy of a classification model, particularly in binary classification tasks. It combines both precision and recall into a single measure, providing a balanced assessment of a model's performance.

Peer-to-peer Correlation

The second category includes the following methods: Peer-to-peer correlation refers to the degree of similarity or correlation between individuals or entities within a peer group or network.

Jaccard coefficient: The Jaccard coefficient is used to find the similarity of the identified clusters C to the true values in P

$$J = \frac{TP}{TP+FP+FN}$$
$$J = \frac{\sum_{ij}\binom{n_{ij}}{2}}{\sum_i\binom{n_i}{2} + \sum_j\binom{n_j}{2} - \sum_{ij}\binom{n_{ij}}{2}}$$

The Rand coefficient is also similar to the Jaccard coefficient although used to measure considering the total data set (accuracy).

$$Rand = \frac{TP+TN}{M}$$
$$Rand = \frac{\binom{n}{2} - \sum_i\binom{n_i}{2} + \sum_j\binom{n_j}{2} - \sum_{ij}\binom{n_{ij}}{2}}{\binom{n}{2}}$$

The Folkes and Mallows coefficient also finds the similarity between the particular clusters generated by particular clustering algorithms as independent markers

$$FM = \sqrt{\frac{TP}{TP+FP} * \frac{TP}{TP+FN}}$$
$$FM == \frac{\sum_{ij}\binom{n_{ij}}{2}}{\sqrt{\sum_i\binom{n_i}{2} * \sum_j\binom{n_j}{2}}}$$

Hubert statistical coefficient

$$\Gamma = \frac{1}{M}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} X_{ij}Y_{ij}$$

Peer-to-peer Correlation includes Jaccard coefficient, Hubert statistical coefficient, Rand coefficient, and Folkes and Mallows coefficient which helps in finding the association between the entities.

Measures Based on Information Theory

Measures based on information theory assess the amount of information present in a system. Key metrics include entropy, reflecting dataset uncertainty, and mutual information, quantifying shared information between variables. Entropy can be considered as the reciprocal of the purity measure to find the degree of disorder among clusters:

$$H = -\sum_i p_i \left(\sum_j \frac{p_{ij}}{p_i} \log \frac{p_{ij}}{p_i} \right)$$

Mutual information is used to measure the reduction in uncertainty in clustering:

$$MI = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{p_i p_j}$$

These measures find utility across disciplines like machine learning and signal processing for optimizing systems and analyzing data. Mutual information quantifies the reduction in uncertainty about cluster assignments when clustering a dataset. By measuring the amount of shared information between data points and cluster labels, mutual information assesses how well clustering reduces uncertainty by revealing underlying patterns or structures in the data.

Optimization Metrics

Optimization metrics in fog computing are essential for measuring the efficiency and performance of distributed computing at the network edge. These metrics enable the assessment and improvement of resource utilization, latency reduction, and overall system optimization in fog-based architectures.

The goals of resource allocation, task scheduling, and workflow scheduling are to maximize the resources of fog nodes by optimizing the job execution process. Resource allocation, task scheduling, and workflow scheduling in fog computing aim to maximize fog node resources by optimizing job execution. Key objectives include efficient task allocation, minimizing delays, and improving overall system performance

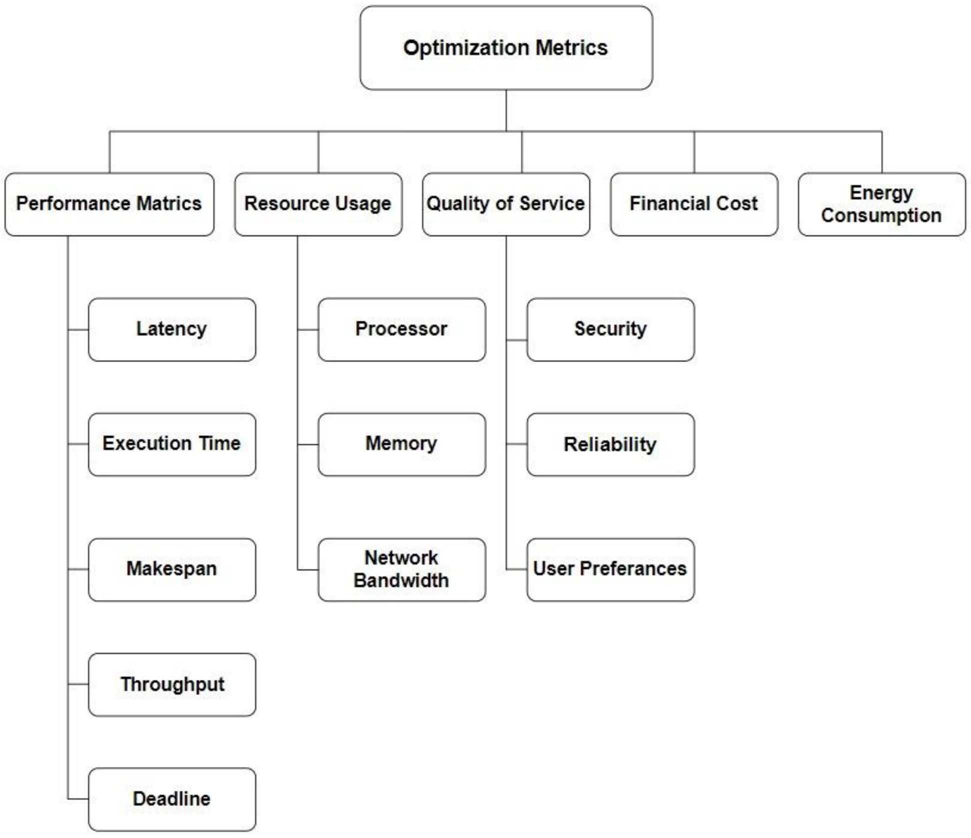


Figure 3.10: Optimization Metrics (Marbukh, 2019)

Figure 3.10 shows optimization metrics such as makespan, latency, throughput, energy consumption, load balancing, and quality of service play a crucial role in achieving these goals. By considering these metrics and employing appropriate algorithms and techniques, fog computing systems can enhance resource utilization, reduce delays, and provide efficient and reliable execution of tasks and workflows.

Optimization metrics in fog computing are critical for achieving optimal performance, resource management, and latency reduction at the network edge. They guide decision-making, ensuring that fog architectures deliver on their promise of efficient and responsive edge computing solutions.

Performance Metrics

Performance metrics in fog computing are vital for evaluating the efficiency and effectiveness of edge computing systems. These metrics provide valuable insights into processing speed, resource utilization, and data transfer rates, enabling the fine-tuning and improvement of fog-based architectures.

Performance metrics in task scheduling for fog computing refer to the parameters used to evaluate and measure the effectiveness and efficiency of the task scheduling process. These metrics provide insights into the performance of the system and help in assessing the quality of the scheduling algorithm or approach. Common performance metrics include makes pan (total time taken to complete all tasks), latency (response time between task submission and completion), throughput (number of tasks completed per unit of time), resource utilization (percentage of resources utilized), and fairness (equitable distribution of resources and workload among fog nodes). These metrics allow for quantitative assessment and comparison of different task scheduling techniques, enabling the selection of optimal approaches for improved performance in fog computing environments. The parameters of performance metrics are as follows:

Latency

One of the most crucial variables for evaluating the effectiveness of any task-scheduling system is latency. Other names for latency include delay and reaction time. The sum of the transmission delay and the computing latency is the total latency.

$$Latency_i = TL_i + CL_i$$

- $Latencyi$: latency,
- TLi: transmission latency
- Computational latency of task ‘i’.

Execution Time

Execution time is the length of time it takes for a system to complete a task. CPU or execution time does not account for the time spent waiting for I/O or other operations to complete.

$$ExeTime_i = FT_i + ST_i$$

- $ExeTimei$: overall execution time,
- FTi: finish time
- STi: the start execution time of task ‘i’.

Make Span

Make span is a key goal of task scheduling that shows how long it will take to execute a process in its entirety.

$$Makespan = CT_l - ST_f$$

- CTl: time when the last task is completed
- STf: starting time of the first task.

Throughput

The number of tasks finished in a system's throughput is measured in units of time.

$$Throughput = \frac{\text{Number of tasks}}{\text{Makespan}}$$

Throughput, in the context of computing and systems, refers to the rate at which tasks or operations are completed or processed within a given time frame. It is a performance metric that measures the efficiency and productivity of a system, indicating how many tasks or units of work are accomplished in a specific period. The measurement of throughput is typically expressed in terms of tasks per second, operations per second, or any other appropriate unit of time. Higher throughput indicates that the system can handle a greater volume of work and is more capable of processing tasks efficiently.

Deadline

The deadline is the amount of time between when a task is submitted and when it must be finished. The completion of each activity at the designated deadline is crucial in real-time applications. Missing a task deadline may be disastrous, especially for difficult real-time applications like air traffic control Jamil (2022). Meeting task deadlines is critical in real-time applications as it ensures timely processing and response. In fog computing, where tasks are distributed across fog nodes, the deadline parameter becomes crucial for efficient task scheduling. The deadline specifies the maximum acceptable delay for task completion, and scheduling algorithms must consider this constraint to ensure tasks are allocated to appropriate nodes that can meet the specified deadlines. Failure to meet task deadlines in time-sensitive applications can have severe consequences, such as compromising safety, system failures, or financial losses. Therefore, in fog computing environments, effective task scheduling algorithms are designed to prioritize tasks based on their deadlines, optimizing resource allocation and ensuring timely task completion within the given constraints.

Performance metrics in fog computing are instrumental in optimizing edge computing solutions. They facilitate informed decision-making, resource allocation, and system enhancements, ensuring that fog architectures deliver the required speed, efficiency, and responsiveness in the evolving digital landscape.

3.10.3 Simulation Setup

For the predictive construct, both supervised and unsupervised machine learning algorithms were used. A simulated environmental setup was constructed to evaluate and appraise the research model being proposed which is SMART FOG system which included an improvised task offloading approach. The experimental environment used is Anaconda Jupiter Python and R. The model is trained through various supervised and unsupervised learning algorithms which include KNN[6], Decision Tree, MLP[7], Logistic Regression algorithm, Lazy IBK, Naive Bayes, SVM[8] being used.

This research work aims to address various aspects of smart fog systems It encompasses a mixed methods approach, combining quantitative analysis and qualitative insights. The research objectives include studying commuting and cloud issues, analyzing technologies for enhancing computation, addressing implementation issues, and identifying the most appropriate machine learning approach for fog computing. The research begins with a comprehensive literature review to identify gaps in existing research and frameworks. Its performance is evaluated using various performance measures, comparing it with existing models. Recommendations and strategies are proposed to overcome these challenges.

[6]K-Nearest Neighbor
[7]Multilayer Perceptron
[8]Support Vector Machine